This Page Is Inserted by IFW Operations
and is not a part of the Official Record

# BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

# IMAGES ARE BEST AVAILABLE COPY.

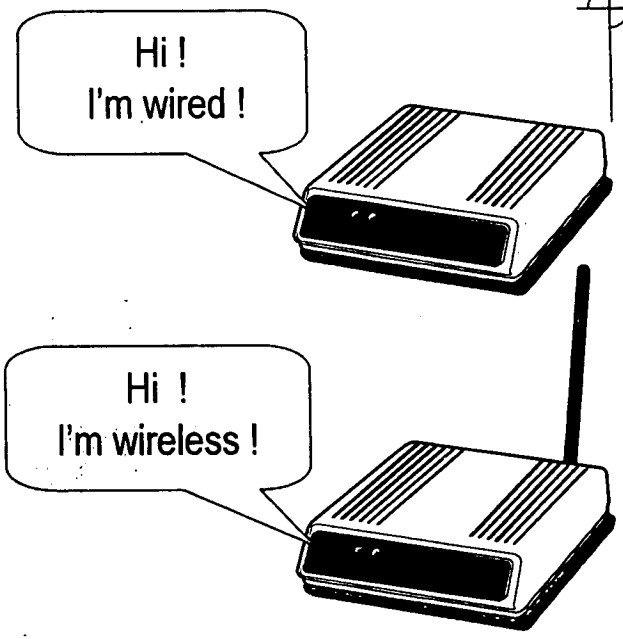
**As rescanning documents *will not* correct images, please do not report the images to the Image Problem Mailbox.**

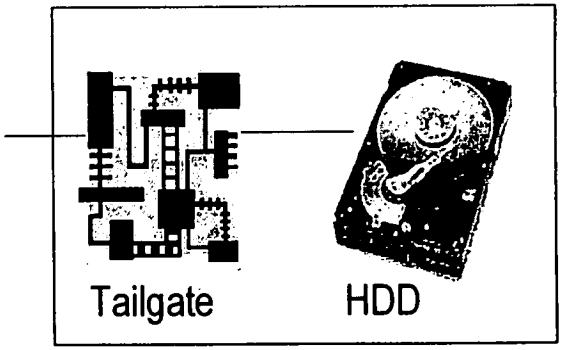
# zetera

January 2003

# What is a μSAN ?

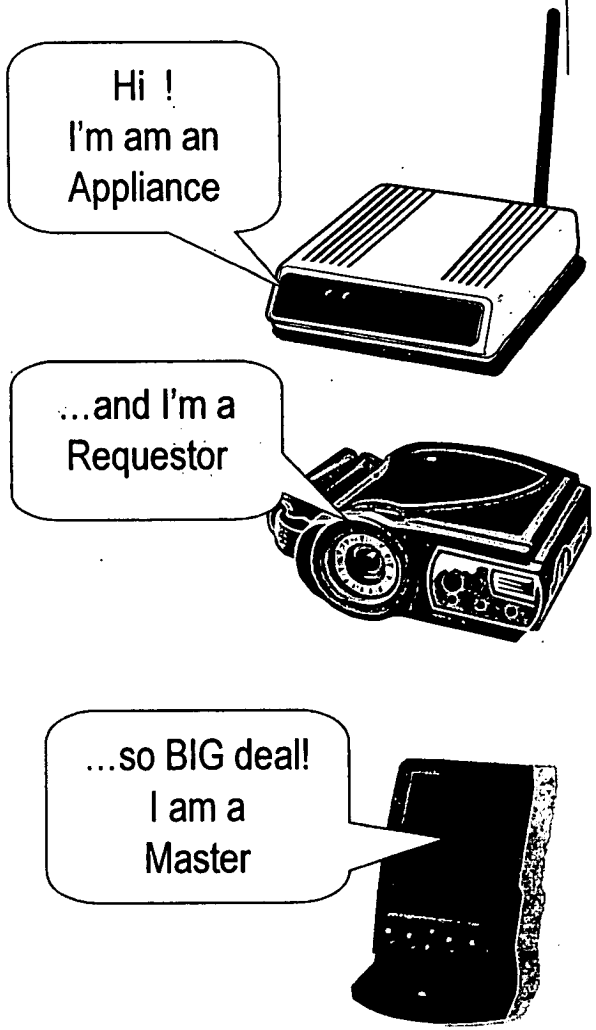
- A μSAN consists of a Storage Device, most likely an EIDE HDD, and a protocol adapter known as a "Tailgate" housed within a suitable enclosure
- The Tailgate adapter is very similar to existing 1394/USB adapters in complexity and cost
- The Tailgate provides the appropriate PHY and protocol level interface either wired or wireless determined by the needs of the product.



# µSAN - Some Basic Concepts

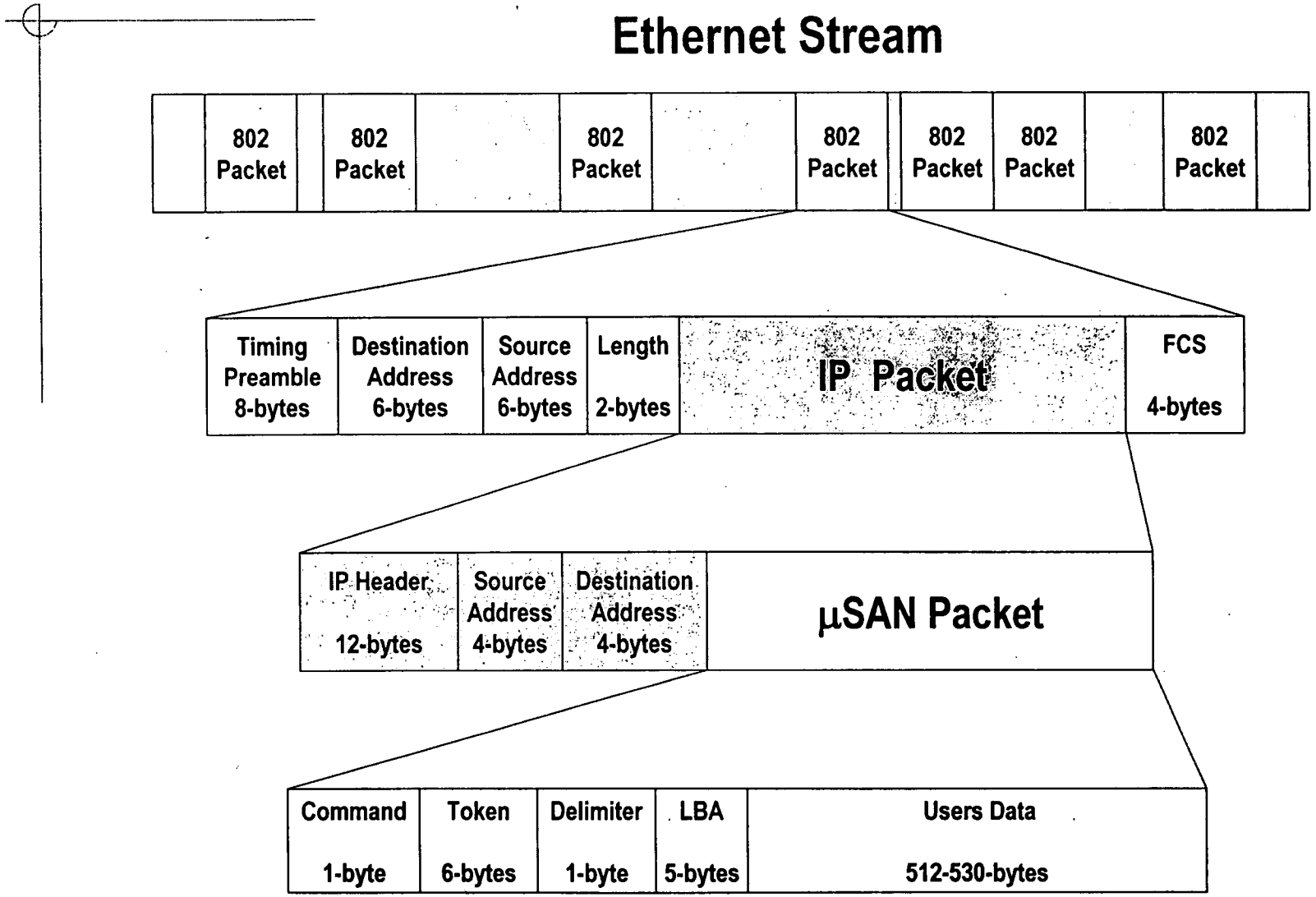
- A µSAN lives in an IP network
  - Ethernet
  - WiFi
  - HomePlug
  - HPnA
  - Any form of IP works
- A µSAN is a Peer on a network of Peers
- A µSAN is not a server – it's an appliance
- A µSAN can work under a master
- A µSAN user is not really a Client
- A µSAN user can be called a Requestor



# How μSAN Works

- Embedding the μSAN protocol inside IP
- μSAN commands
- Discovering a networked μSAN device
- Owning your own storage volume
- Sharing data on a μSAN
- Security and authentication
- Spanning and redundancy
- Dissagregation - Real life examples

# Embedding the μSAN protocol inside IP



# μSAN Broadcast IP Protocol

- μSAN uses Broadcast IP Protocol in conjunction with UDP Transport Protocol to perform two functions:
  - Name Resolution
  - μSAN Find (discovery)
- A "found" or discovered μSAN will identify the IP address of its Root to a requestor using a Unicast response directly to the Requestor.
- The Requestor must wait an appropriate length of time to ensure that all FIND RESPONSES are received and may want to retry to ensure none are lost.
- In true Peer to Peer fashion, the root has no name.

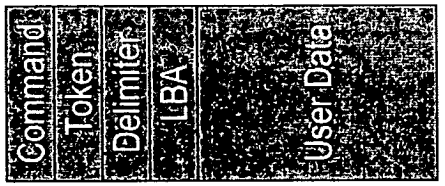
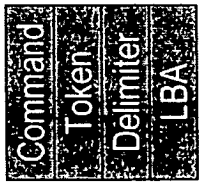
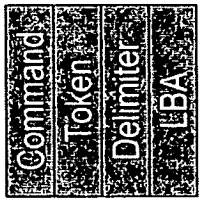
# µSAN Broadcast Protocol

| | | |
|---|---|---|
| **µSAN Find Broadcast** |  1B | This request is issued by the Requester looking for the root IP of all µSAN on the LAN. Every µSAN that receives this packet must respond with µSAN Find Response packet to the calling IP/UDP Port. The µSAN Find packet is addressed to UDP Port XX |
| **µSAN Find Response** |  1B 4b | Each µSAN in response to a µSAN Find packet responds to the Requester with its IP address in a µSAN Find Response packet. This packet is addressed to the requesting IP/UDP Port. It is understood that there may be many responses to the request issued on the network and will be arbitrated at the network layer |
| **Name Resolution Request** |  1B 16B | The µSAN Requester issues this packet to UDP Port XX as a request for Name to IP resolution. The µSAN that recognizes this Name as one of its partitions responds with a Name Resolution Response Packet. |
| **Name Resolution Response** |  1B 16B 4b | The µSAN issues this packet in response to the Name Resolution Request packet with the IP address associated with the Name. It is issued to the requesting IP/UDP Port. The name is resent in case of multiple requests and the "Sorcerer's Apprentice Syndrome" present in UDp |

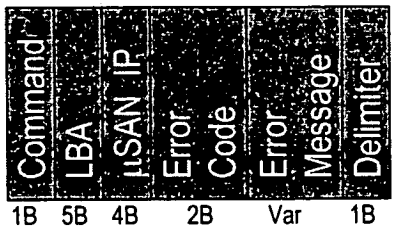
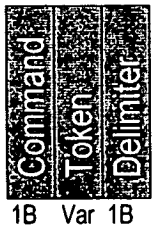
# Basic μSAN Commands

- μSAN Block Transfer Protocol is a master/slave architecture with the Requester as the master
- Operates under either UDP or TCP transport protocol and defaults to the most efficient mode
- Commands are variable length depending on function
- The block size is fixed at 512 (or 530) bytes
- LBA (Logical Block Address) field is 40 bits - 512 Tbytes

# Basic μSAN Commands

| | | |
|---|---|---|
| **Transfer Command** |  | This command is used to transfer the data either as a write to the μSAN or the result of a request from the μSAN. One block of data is transferred. In a write from Requester operation, this is the only command that is transferred from the Requester. The μSAN responds with an ACK Command. This command may be sent to either unicast or multicast destination IP addresses. |
| **Request Command** |  | This command is a request for a transfer from the μSAN. The μSAN responds with a Transfer Command containing the requested data. No ACK Command is required in response to this transfer. This command may be sent to either unicast or multicast destination IP addresses. |
| **Request Lock Command** |  | The function of this command is to provide for semaphore functionality of the data. This is a transfer request from the μSAN, however, it locks out all other Requestors. The lock is released when a Transfer Command is received from this Requester or a time-out occurs.. This command may be sent to either unicast or multicast destination IP addresses |
| **Ack Command** |  | This command acknowledges a successful transfer. It is required if communication via UDP to acknowledge a successful transfer from a Requester to the μSAN. In TCP, a Requester may wait for an ACK to ensure single threaded error handling. The LBA that was received is returned avoiding the "Sorcerer's Apprentice" problem inherent in IP due to time-out retry. The μSAN IP address is the address of the μSAN that responded to the transfer. |

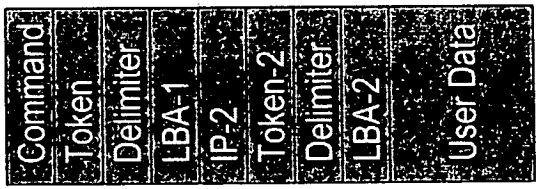
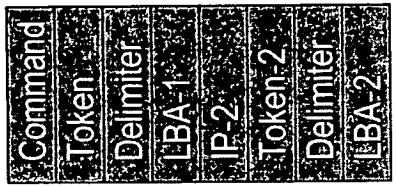
# Basic μSAN Commands

| | | |
|---|---|---|
| **Error Command** |  | The Error Command indicates that an operation could not be completed for some reason and is a transfer from the μSAN to the Requester. The μSAN IP address represents the address of the μSAN that responded to the transfer. (For example, a multicast GO TRANSFER command to a set of mirrored partitions may generate an ERROR from one of the μSAN back to the transferring μSAN who would then pass that ERROR back to the client for exception resolution. The ERROR that is passed back to the Requester will have the destination's μSAN IP address.)<br><br>Error Codes (partial list):<br>01h Invalid Authorization<br>02h Partition has locked you out<br>04h Go Command - Invalid Authorization at destination<br>08h Go Command - locked Partition at the destination<br>10h LBA is out of Range<br>20h LBA is Write Protected |
| **Release Partition Command** |  | This command will cause the μSAN to release a partition, erase the data blocks within that partition, and release the IP/Name. The released blocks are placed back into the root pool for future allocation. |

# μSAN Extended- "Go" Commands

- Specifically used to copy a logical block from one μSAN partition to another μSAN partition.
- Initiated by the Requester who resolves both of the partition's Names to their relative IP address and passes the command to one of the μSANs.
- The result is to double the effective bandwidth and to remove a burden from the Requester.
- "Go" commands are not necessary for basic μSAN communication and are considered optional

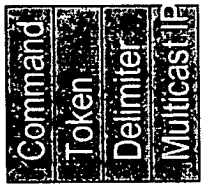
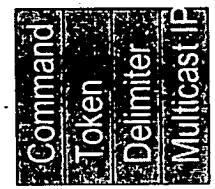
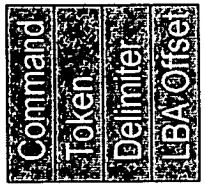
# µSAN Extended- "Go" Commands

| | | |
|---|---|---|
| **Go Transfer Command** |  | The Go Transfer Command causes an addressed µSAN to go active on the network and autonomously transfer a block of data to another µSAN identified in "IP-2" |
| **Go Request Command** |  | The Go Request Command is similar to the Go Transfer Command. The Requester issues this command to a µSAN to have it request data from another µSAN. At the completion of the transfer, the µSAN will ACK or ERROR the Requester. TCP transport is used to send the request to the second µSAN. |
| **Go Request Lock Command** |  | The Go Request Lock Command is similar to the Request Lock command in the basic command set except that it requests an addressed µSAN to lock access to another µSAN identified in "IP-2" |

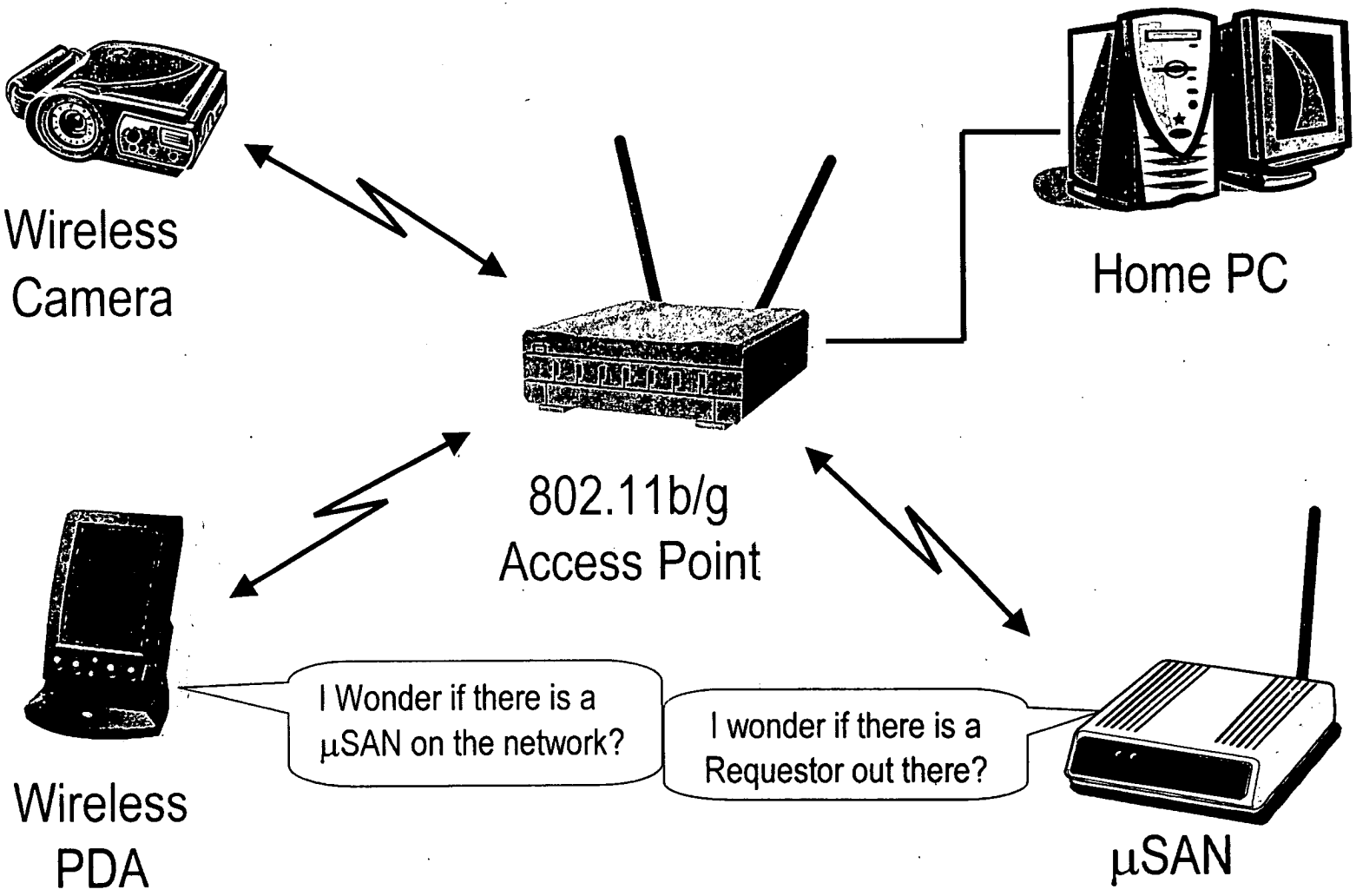
# μSAN Commands with Multi-Cast IP

- Optional command set specifically used to support Spanning and Mirroring among multiple μSANs.
- Requires Requestors as well as μSANs and routers to support Internet Group Management Protocol (IGMP).
- The burden on Requestors and μSANs to support IGMP is light, but support on routers is much heavier.
- These commands allow the μSAN to exploit the unique capabilities of Multi-Cast to provide Spanning and Mirroring.

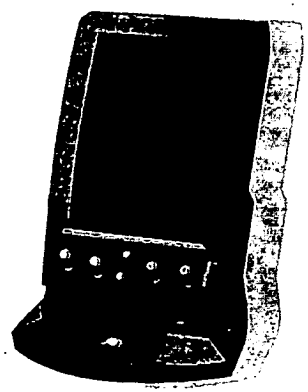
# μSAN Commands with Multi-Cast IP

| | | |
|---|---|---|
| **Set Multi-Cast IP Command** |  | This command will cause the μSAN to cause the partition to enable multicast capability to this partition at the specified multicast IP address. Based upon the physical network interface, the μSAN will translate this address appropriately to the multicast MAC. (Note: It may be possible for a μSAN to support more than one multicast address.) Upon receipt of this command, the μSAN will issue a IGMP JOIN message and will respond to IGMP QUIRES to this multicast address. |
| **Release Multicast IP Command** |  | This command will cause the μSAN to cause the partition to release the multicast IP associated to this partition. (Note: It may be possible for a μSAN to support more than one multicast address.) The μSAN will no longer answer multicast IP to this address nor will it respond to an IGMP QUERY. |
| **Set LBA Offset Command** |  | This command though used in multicast, is independent of multicast status. This command will set the LBA relative starting address for the partition (used in spanning). |

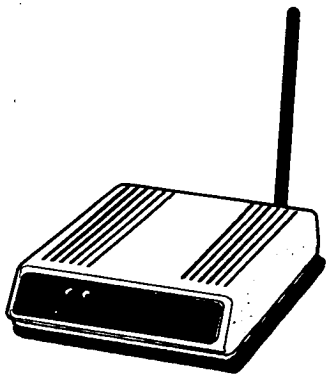
# Discovering a networked μSAN device



# Discovering a networked μSAN device



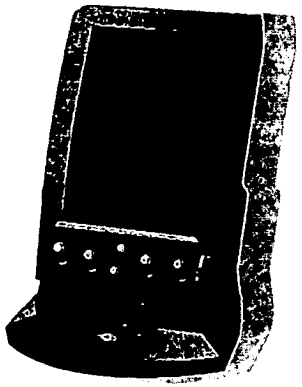
**Send Find μSAN Broadcast Command = 80h**



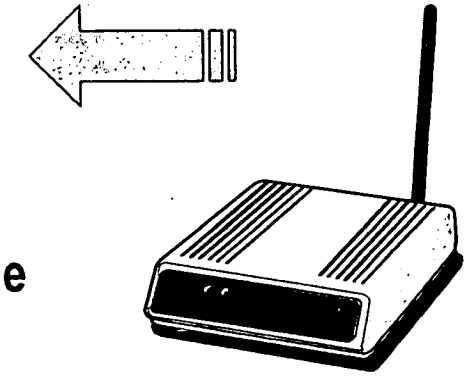
**I'm Looking for a μSAN**

**I'm Looking for a μSAN**

# Discovering a networked μSAN device



**Send μSAN Find Response**
**Command = 81h**
**IP Addr =192.29.100.0**

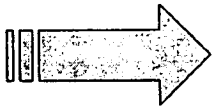


**μSAN 192.29.100.0**

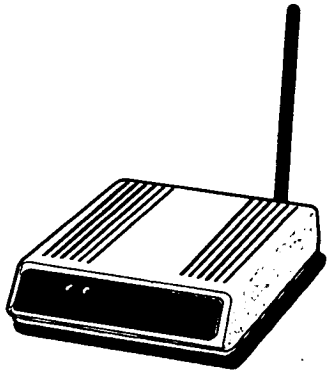
**I Hear you**
**I'm here**

# Owning a μSAN storage volume





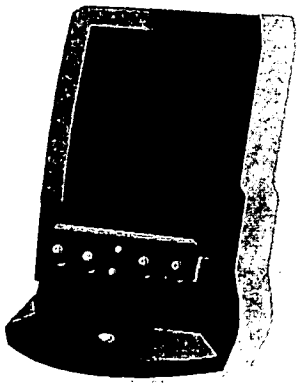
Get μSAN Root Status
(Request Transfer Command)
LBA = 0
Token = (don't care)



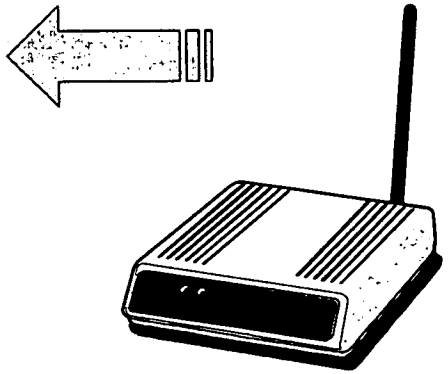
μSAN 192.29.100.0

How much Free Space
is available ?

# Owning a μSAN storage volume



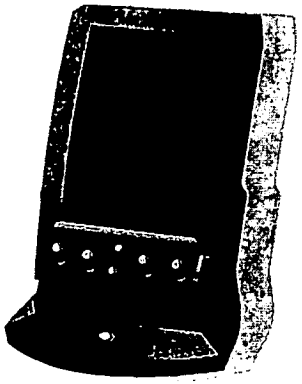
a. Version
b. Total Capacity
c. Available Capacity
d. Speed
e. Reliability
f. Portability
g. QoS Capability
h. ...

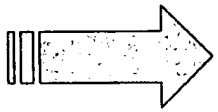


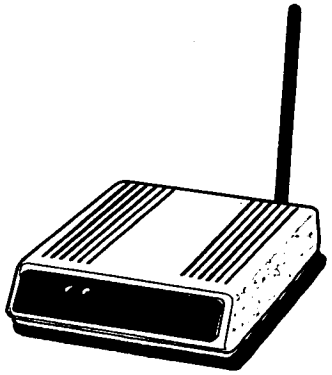
μSAN 192.29.100.0

I have lots of space
Here's my root status

# Owning a μSAN storage volume





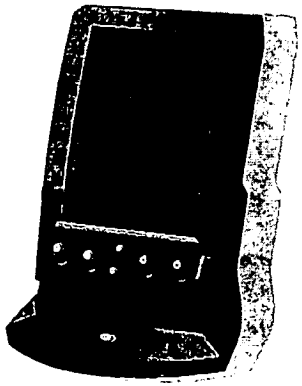


μSAN 192.29.100.0

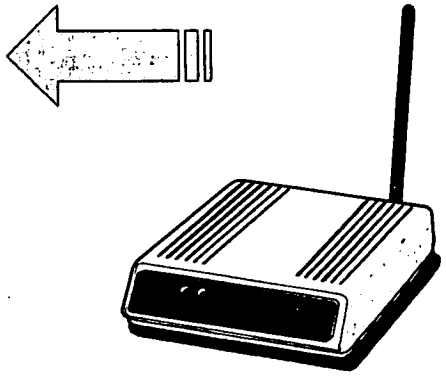
OK!
Reserve a partition

**Allocate a Partition**
**(Block Transfer Command)**
- **LBA = 0**
- **Token = (don't care)**
- **Data bytes 0 – 511**
  - **a. Partition Name = PDA**
  - **b. Partition Token= PDA's token**
  - **c. ID Character String**
  - **d. Authentication Tags**
  - **e. Partition Size**
  - **f. Personality Tags**

# Owning a μSAN storage volume



ACK

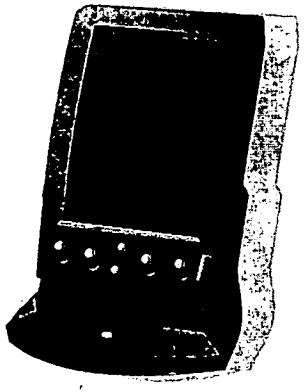


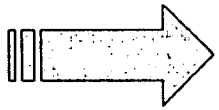
**μSAN 192.29.100.0**

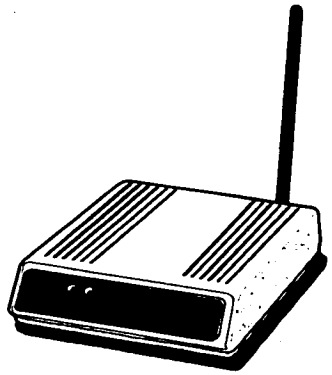
**OK!**

**PDA's Partition is Reserved**

# Owning a μSAN storage volume







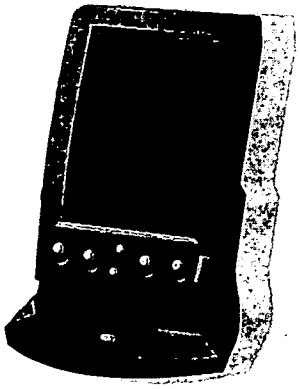
**Write a Block of Data**
**(Block Transfer Command)**

**LBA = 1000**
**Token = PDA's token**
**Data bytes 0 – 511**
**(PDA's data)**

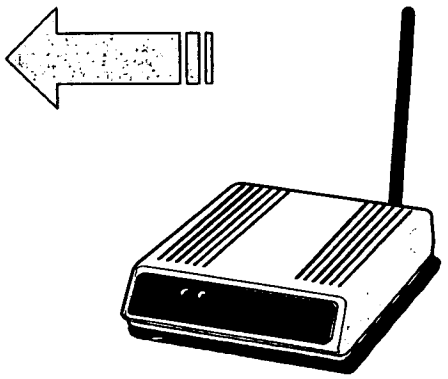
**Hey PDA Partition!**

**I want to write some data**

# Owning a μSAN storage volume



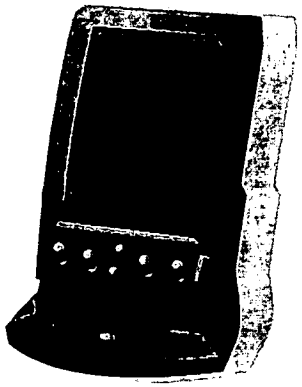
ACK

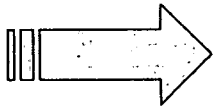


**PDA Partition**

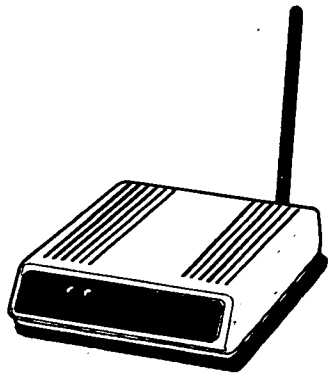
**OK**
**Data Block has been written**

# Owning a μSAN storage volume





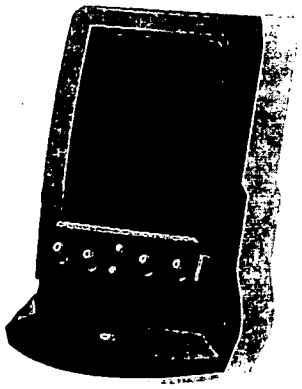
Read a block of data
(Request Transfer Command)



LBA = 1000
Token = PDA's token

Hey – PDA Partition!
I want to read back my
data to verify it

# Owning a μSAN storage volume

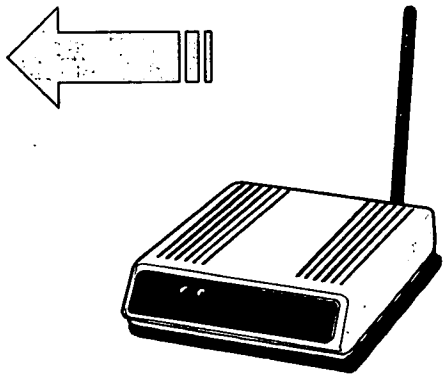


**Block Data Transfer**

**Block Address = 1000**

**Data Bytes 0-511**

**(PDA's Data)**

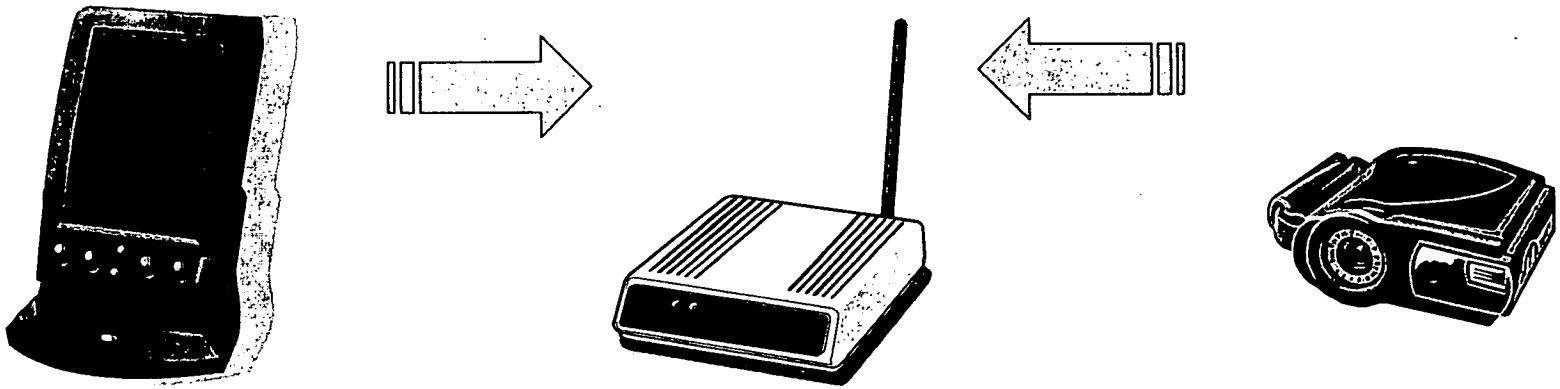


**PDA Partition**

**OK!**

**Here is your Data**

# Sharing data on a μSAN



Hey! Camera

I want to see the pictures you've taken

Hi PDA !

I save them in μSAN 192.29.100.0

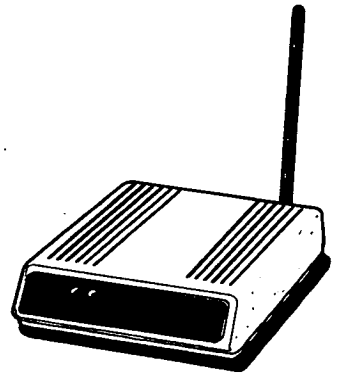
Here's my token

And Partition Name

# Owning a μSAN storage volume







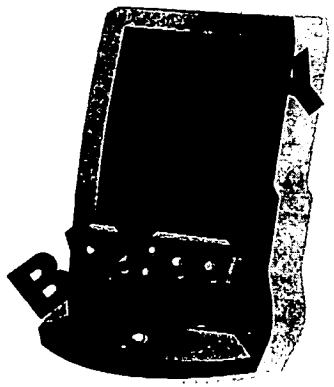
**Read a Block of Data**
**(Request Transfer Command)**

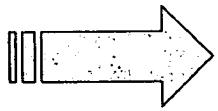
**LBA = 1000**
**Token = Camera's Token**

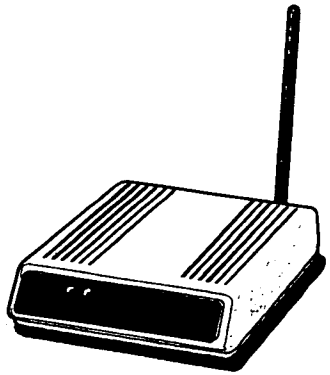
**Hey Camera Partition!**

**I Want to see pictures**

# Security and authentication







Hey PDA Partition!

I Want to erase data.

(chuckle)

Write Data Block

(Block Transfer Cmd)

Requestor IP = Wrong

Requestor MAC = Wrong

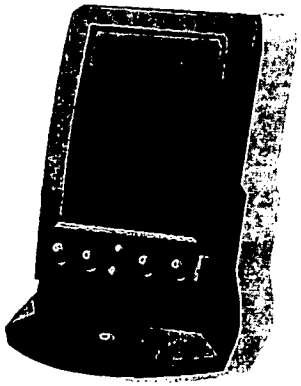
LBA DAC = Wrong

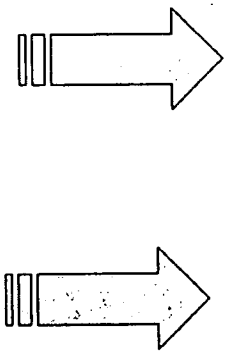
Payload DAC = Bogus

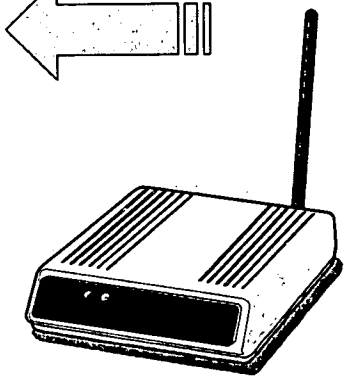
Not even close!

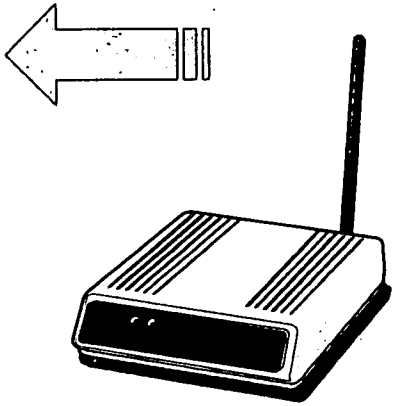
No Response

# Spanning Partitions with Multicast









**My data will not fit on one μSAN**

**2 M Blocks!**

**I'll use Multicast**

**Writing blocks**

**1 – 1 M**

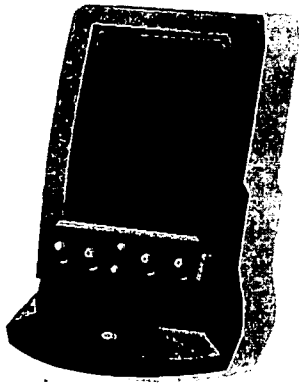
**These blocks are mine so I will write them and ACK**

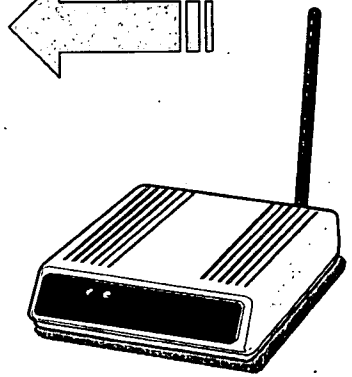
**Writing blocks**

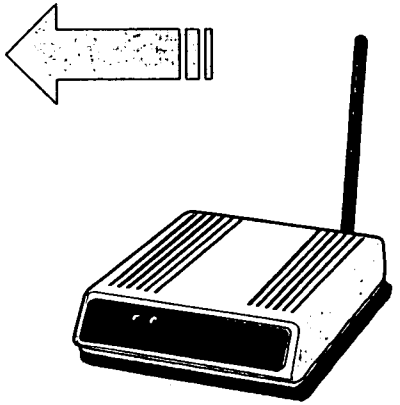
**1 M +1 – 2M**

**These blocks are mine so I will write them and ACK**

# Redundant Partitions with Multicast







**My data is very valuable!**

**I can't risk loss or corruption**

**I'll use Multicast**

**Writing blocks**

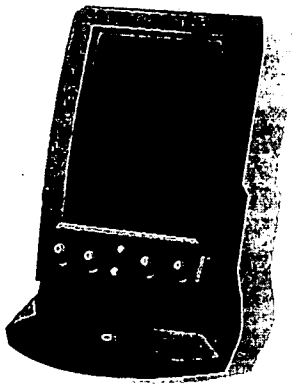
**These blocks are mine so I will write them and ACK**

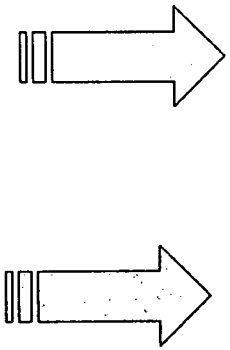
**Writing blocks**

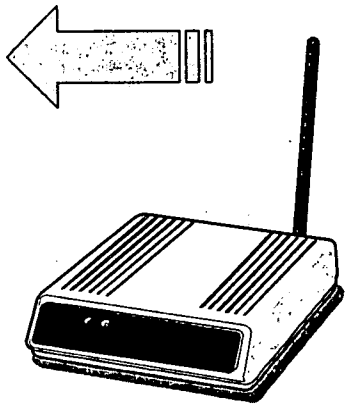
**(Mirror)**

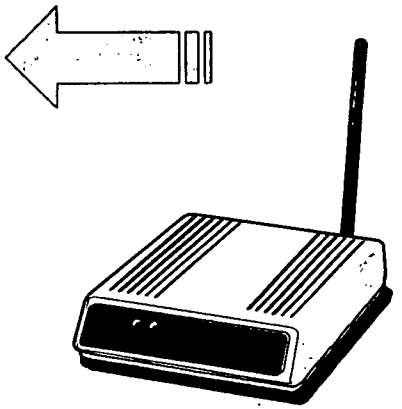
**These blocks are mine so I will write them and ACK**

# Striping Data with Multicast









**My data is very valuable!**

**I can't risk loss or corruption**

**I'll use Multicast**

**Writing blocks**

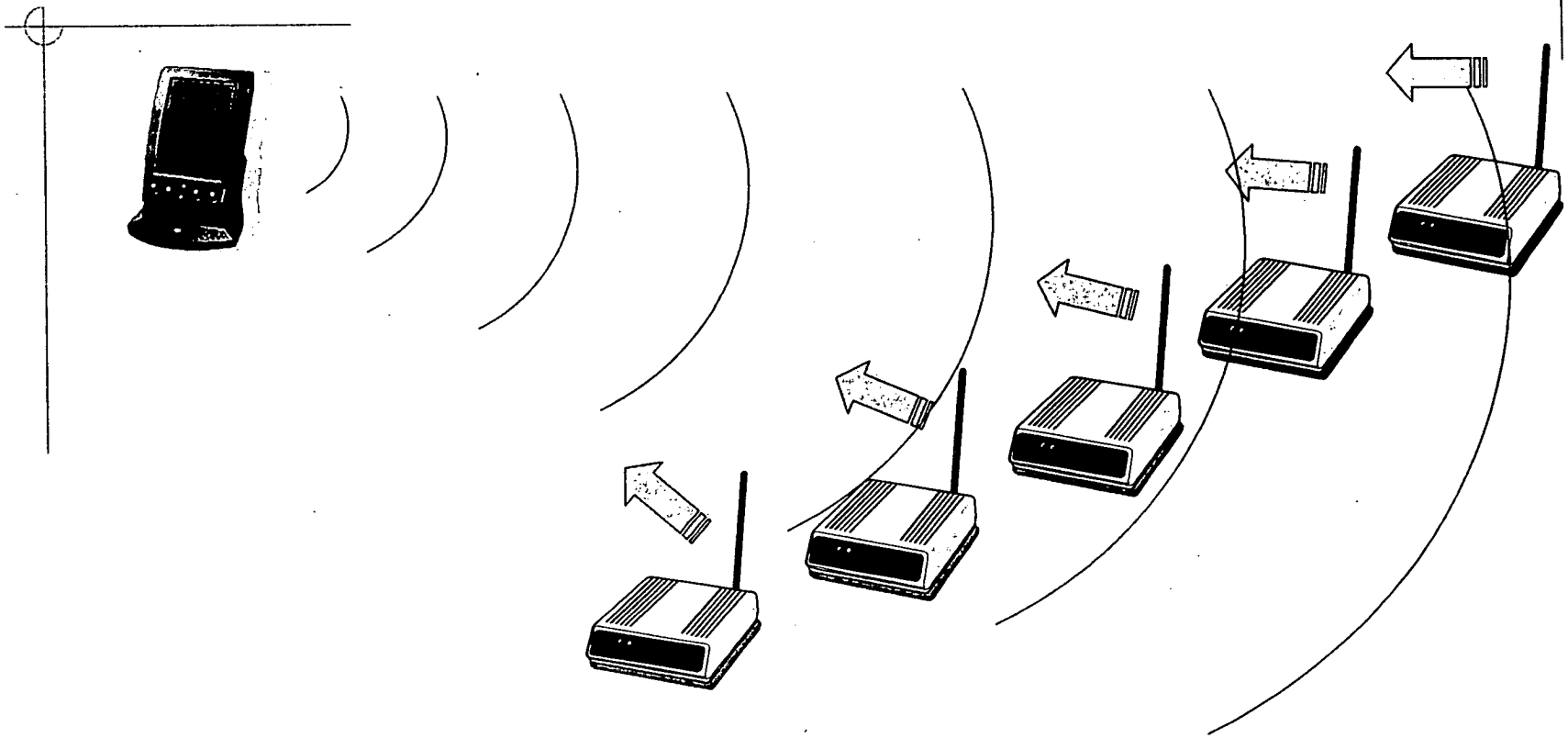
**These blocks are mine so I will write them and ACK**

**Writing blocks**

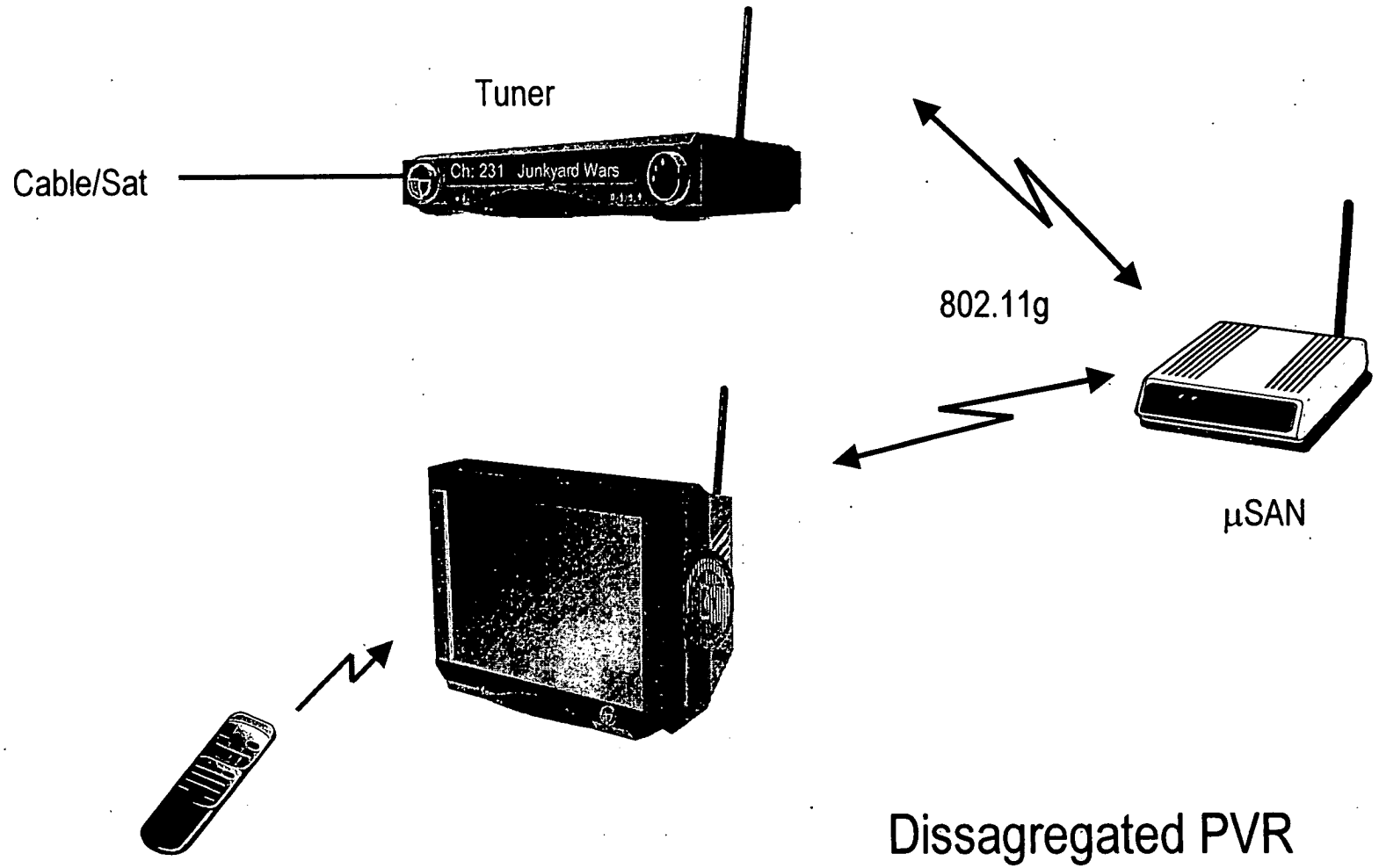
**(Mirror)**

**These blocks are mine so I will write them and ACK**

# RAID with Multicast

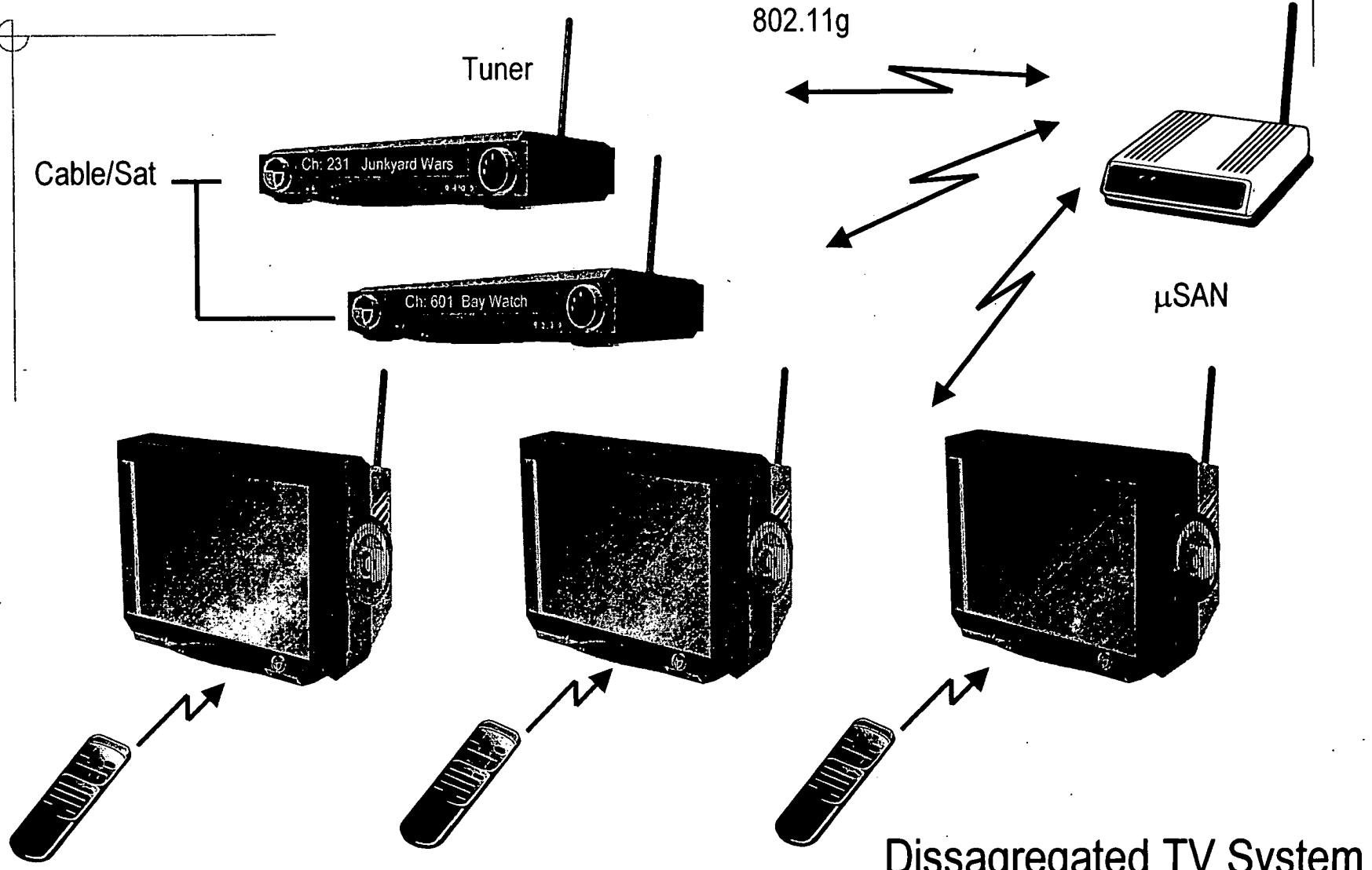


# Dissagregation - Real life examples




Dissagregated PVR

# Dissagregation - Real life examples




Dissagregated TV System

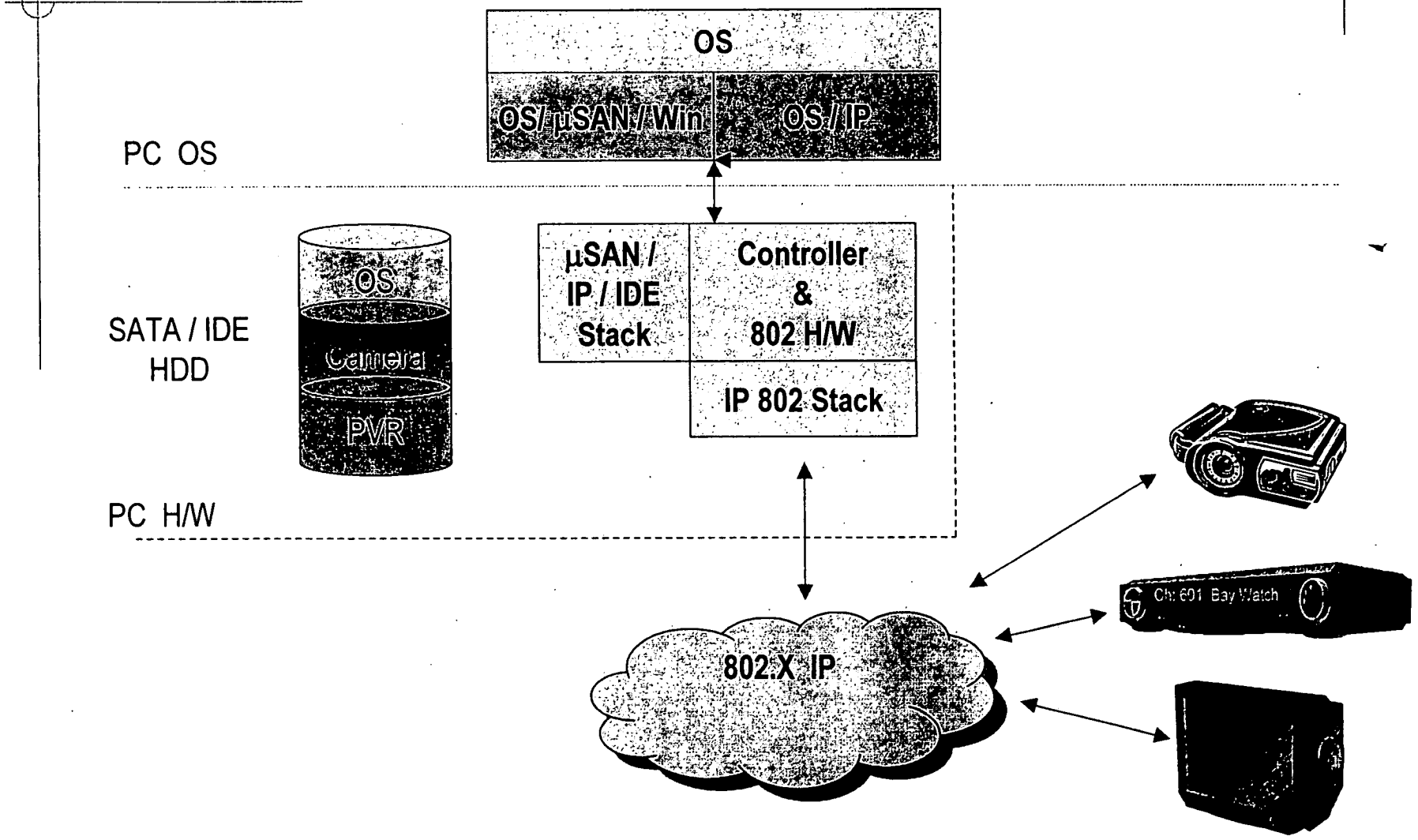
# Competitive Technologies – IP

| | µSAN | NAS | iSCSI |
|---|---|---|---|
| Connection | Internetwork Protocol | Internetwork Protocol | Internetwork Protocol |
| Data Access Method | Block Level | File Level | Block Level |
| File Format | None imposed. Client defines | Industry standards i.e. FAT16, FAT32, NTFS, HPFS, Unix, Solaris ... | None imposed. Client defines |
| O/S Support | Agnostic – all supported | Support defined by OS compatibility options | Agnostic – all supported |
| Master requirement | None required | NAS is a master | Master required on each subnet |
| Control Processor | 32 bit Low-Cost Embedded RISC | Pentium Class or greater | Hi-performance RISC with H/W acceleration (TOEs) |
| DHCP Requirement | None for simple networks. DHCP on local net or internal for complex networks | Contains DHCP services | Relies on Master |

# Competitive Technologies - IP (con't)

| | µSAN | NAS | iSCSI |
|---|---|---|---|
| Multiple Volumes (Partitions) | Real Volumes – Created, used, shared and deleted by client with ownership rights | Supported as volumes or extensions defined by the OS | Virtualized Volumes – Supported through master with ownership rights |
| Volume Addressing | Volumes are addressed by a unique IP address | Virtual Volume support only – Directroy structured | Partitions are accessed by a master controller using a directory lookup service |
| Number of volumes | No fixed limit | No fixed limit | No fixed limit |
| Discovery | Supports peer discovery, UPnP and DNS depending upon environment | OS dependent – NetBIOS, DNS, Master Brower | iSNS – ISCSI master Name Server |
| Authentication | Supports light, efficient and secure rotating key authentication across both private IP networks and through the Internet | OS supports many forms of authentication including Kerberos, PKI, IPSec, etc. on both private IP networks and through the Internet | Relies on heavy IPSec security services from connected masters on private IP networks |
| Controller cost | $10 - $15 | $100+ | $500+ |

# Embedded ubiquitous µSAN



# EXHIBIT A – US 60/441739